Bibliotec: Enhancing Literary Engagement through Synchronized AI-Narrated Audio and Text

1: INTRODUCTION

Abstract -**Contemporary reading habits face unprecedented challenges as high-stimulus digital platforms reshape cognitive engagement, often resulting in shortened attention spans and reduced reading frequency. Although audiobooks offer an alternative, existing formats frequently lack the immersive qualities necessary to sustain modern users' attention. This project proposes an interactive digital audio library Bibliotec that synchronizes AI-generated narrated audio with corresponding textual content to enhance engagement with literary works. By integrating advanced Text-to-Speech (TTS) models capable of delivering naturalistic and contextually expressive narration, the platform overcomes limitations of conventional monotone TTS solutions (Mu, Yang, & Dong, 2021). Users can seamlessly follow highlighted text while listening, creating a dual-modal reading experience that caters to diverse cognitive preferences. The system's distraction-free interface, minimal hardware requirements, and reliance on public domain texts ensure broad accessibility and ethical compliance. Preliminary evaluations indicate that combining synchronized text and audio can improve sustained attention and reading satisfaction. In doing so, this project contributes a novel approach to addressing the decline in book reading by employing AI-driven, user-centric design principles, thereby aligning modern reading practices with evolving digital media consumption patterns.**

2: THE PROBLEM: A CRISIS IN LITERARY ENGAGEMENT

The decline in book reading, driven in part by shifts in dopamine responses to high-intensity digital media (Volkow et al., 2011 ; Wilmer, Sherman, & Chein, 2017), underscores a need for innovative reading platforms that sustain engagement. In response, this project, Bibliotec, merges AI-generated narrated audio with corresponding text in real time, creating a multisensory reading environment. By offering continuous stimulus in the form of synchronized audio and text, Bibliotec aims to revitalize interest in literature, particularly in an era marked by shortened attention spans.

A key component of this platform is its advanced Text-to-Speech (TTS) subsystem, which addresses the limitations of existing TTS solutions, often monotonous and detached from the emotional context of the text (Mu, Yang, & Dong, 2021). This subsystem employs an N-dimensional emotion-space framework), to deliver naturalistic, emotionally expressive narration. This approach not only aligns with user accessibility goals but also caters to the modern brain's demand for continuous, engaging stimuli.

3. THE SOLUTION: BIBLIOTEC'S VISION

Given Bibliotec's commitment to an immersive, flow-state reading experience, the TTS engine must accommodate gradual transitions, subtle inflections, and contextual variations in emotion. To address these challenges, the project adopts a continuous N-dimensional emotion model that extends beyond basic emotion tags to capture fine-grained prosodic variations (Rabiee et al., 2019 ; Mu, Yang, & Dong, 2021). This model influences specific acoustic parameters such as pitch, speech rate, intonation contours, and articulation clarity, ensuring that audiobook narration remains authentic and engaging for users (Yildirim et al., 2004 ; Lee et al., 2004).

In developing this approach, I previously attempted to build a machine learning model that analysed variables such as screen time, location, biometric data, sleep history, and user biology.

That model produced outputs for arousal and valence over time to predict mental health, and it served as an inspiration for mapping tone from textual content. Although the initial model focused on two dimensions, arousal and valence (Russell, 1980 ; Russell, 1980), I plan to expand this framework to include additional dimensions, allowing for a more nuanced and precise control over emotional delivery in narration (Rabiee et al., 2019 ; Mu et al., 2021).

4. EXPRESSIVE TEXT-TO-SPEECH (TTS) ARCHITECTURE

A key innovation of the Bibliotec platform lies in its expressive Text-to-Speech (TTS) engine, designed to overcome the limitations of traditional monotone narration by enabling nuanced emotional expressiveness (Mu, Yang, & Dong, 2021). This system is central to delivering an engaging, immersive auditory experience that dynamically responds to the narrative's emotional contours. To achieve this, Bibliotec employs an N-dimensional emotion-space model, where each axis corresponds to a prosodic or affective feature of human speech (Rabiee et al., 2019 ; Mu et al., 2021). Unlike conventional TTS frameworks that rely on fixed emotion categories, this model allows for smooth transitions, mixed affect, and subtle tonal variations—much like real human expression (Scherer et al., 2003 ; Batliner et al., 2006).

4.1 Categorical vs. Continuous Emotion Models

Traditional TTS systems have historically relied on categorical emotion models, labelling speech segments as "happy," "sad," "angry," or "neutral" (Batliner et al., 2006). While straightforward, this approach tends to produce rigid and stereotyped vocal expressions that fail to capture the nuanced flow of human emotion within narrative text. These categorical systems often ignore the fluid interplay of emotions that naturally occur in storytelling—such as shifts from calm introspection to moments of urgency or joy.

In contrast, Bibliotec introduces a continuous emotion model, allowing the system to generate speech that adapts fluidly across multiple emotional states (Rabiee et al., 2019 ; Mu et al., 2021). This approach draws inspiration from dimensional psychological theories of emotion (e.g., Russell's circumplex model) (Russell, 1980 ; Russell, 1980), extending beyond the common two axes of valence and arousal to incorporate a wider set of prosodic and affective dimensions relevant to expressive narration (Scherer et al., 2003 ; Fontaine et al., 2007).

4.2 The N-Dimensional Emotion Space

The N-dimensional emotion space serves as the backbone of Bibliotec's expressive TTS pipeline. Each dimension within this space corresponds to a specific acoustic or prosodic feature that contributes to emotional tone and narrative delivery (Yildirim et al., 2004 ; Lee et al., 2004). These include—but are not limited to—pitch, speech rate, volume, articulation clarity, intonation contour, shimmer, and jitter (Scherer et al., 2003 ; Eyben et al., 2016). By decomposing emotion into these low-level acoustic components, the system is able to interpolate between emotional states with precision.

For instance, an emotionally complex passage might be rendered with a gradual increase in pitch and intensity (reflecting rising excitement), followed by a tapering off in speech rate and volume to indicate a return to calm. This kind of modulation is made possible by treating speech synthesis as a function of movement through this multi-axial space, rather than as the selection of a single label.

4.3 Emotional Parameters & Narrative Implications

The following table outlines a subset of the emotional dimensions and their corresponding speech parameters. Each dimension is associated with specific narrative implications and practical examples, illustrating how the system maps emotional content to acoustic expression (Scherer, 1986 ; Juslin & Laukka, 2003)

Dimension	***Speech Parameter***	***Narrative Implication***	***Example***
Valence (Positive/Negative)	Pitch (Hz)	Higher pitch for positivity, lower for negativity	Rising intonation in joyful scenes
Arousal (High/Low Energy)	Speech Rate (WPM)	Faster for excitement, slower for calmness	Quick delivery during action sequences
Intensity (Subtle/Strong)	Volume (dB)	Higher amplitude for strong emotions	Louder emphasis during climaxes
Polarity (Affirming/Doubting)	Intonation Contour	Rising pitch for uncertainty, falling for assertiveness	Modulated pitch in questioning phrases
Control (Dominant/Submissive)	Articulation Clarity	Clearer articulation for confidence, softer for passivity	Crisp enunciation in authoritative speech
Tension (Relaxed/Tense)	Jitter & Shimmer	Perturbations for nervousness, smooth flow for calmness	Slight variability in anxious moments
Narrative Style (Descriptive/Dynamic)	Prosodic Variation	Greater vocal variety for dramatic storytelling	Varied prosody for engaging narratives

Fig. 4 presents this table visually how narrative elements and emotional tone translate into vocal delivery parameters. These parameters are not isolated; they interact dynamically. For example, a state of high arousal may influence both speech rate and intensity, while shifts in valence can co-occur with changes in intonation and articulation clarity. This interdependency enriches the naturalism of the output speech.

4.4 Mapping Text to Emotion Vectors

To accurately capture emotional nuance in narration, Bibliotec processes each spoken text segment—ranging from n-grams (unigrams, bigrams, trigrams) to natural language phrases (Pitrelli et al., 2006)—and maps them to vectors in an N-dimensional emotion space (Rabiee et al., 2019). This mapping is achieved through a multi-stage analytical pipeline that extracts both semantic meaning and acoustic properties from aligned audio and text data.

Step 1: Temporal Alignment of Words and Audio

The process begins with the alignment of text and audio using Montreal Forced Aligner (MFA; McAuliffe et al., 2017). MFA generates TextGrid files, annotating each spoken word with precise start and end times within the corresponding audio file.

From these TextGrid files, each word is extracted along with its timestamp. This enables the construction of temporally accurate segments for further analysis. The system supports both fixed-length overlapping n-grams and natural language phrases, ensuring flexibility in emotional modelling across different textual granularities.

Each aligned audio file (typically in .wav format) is loaded using the Pydub library (Robert & Webber, 2011). For each segment (whether n-gram or phrase), a corresponding audio snippet is extracted based on its time range. These segments are subsequently processed to compute a rich set of acoustic and semantic parameters.

The resulting data for each segment includes the following values, stored in a CSV file:

- words_spoken,
- acoustic_valence,
- semantic_valence,
- arousal,
- speech_rate,
- intensity_db,
- polarity,
- control_articulation,
- tension_jitter,
- tension_shimmer,
- narrative_variation

Each of these values corresponds to a dimension in Bibliotec's N-dimensional emotion space, supporting precise and expressive conditioning of the TTS system. This structured, multi-layered representation ensures that emotional tone can be learned and synthesized with nuance across both textual and auditory modalities.

Step 2: Segment Generation: N-Grams and Phrases

With word-level timestamps available, the system constructs overlapping n-grams (e.g., unigrams, bigrams, trigrams) as well as semantically coherent phrases. These units represent contiguous or meaningful sequences of words. For each segment, a matching audio snippet is extracted and saved. These segments serve as self-contained emotional and prosodic units for further analysis.

Step 3: Semantic Valence Extraction

To assess the emotional connotation of each text segment, the system employs SentiWordNet (Baccianella, Esuli & Sebastiani, 2010) in conjunction with the NLTK toolkit (Bird, Klein & Loper, 2009). Each word is lemmatized and POS-tagged; stop words are removed. The remaining words are matched with their sunsets in WordNet to retrieve sentiment scores. The semantic valence for a segment is calculated as the mean sentiment score across its valid tokens.

Step 4: Acoustic Feature Extraction

Each segment's audio snippet is then processed to extract its prosodic and emotional features using librosa, praat-parselmouth, and custom signal processing scripts. These features are directly mapped to the emotional dimensions in the Bibliotec model:

- Acoustic Valence: Average pitch (via librosa.pyin)
- Semantic Valence: From SentiWordNet-based analysis
- Arousal: Root Mean Square (RMS) amplitude
- Speech Rate: Words-per-minute based on timestamp intervals
- Intensity (dB): RMS converted to decibel scale
- Polarity: Slope of the pitch contour
- Control (Articulation Clarity): Spectral centroid and formant clarity
- Tension Jitter: Measured through short-term pitch perturbations
- Tension Shimmer: Derived from amplitude modulation irregularities
- Narrative Variation: Pitch standard deviation over the segment

Step 5: Emotion Vector Construction and Export

Each processed segment is assigned an emotion vector.

These vectors are exported to structured CSV files alongside their audio segments, creating a rich dataset for training the TTS engine. By conditioning the TTS output on these vectors, Bibliotec achieves highly expressive, context-aware narration.

Step 6: Mel Spectrogram Conversion

Following the extraction of emotional features, each audio segment undergoes a transformation into a mel spectrogram—an essential representation for training expressive speech synthesis models. This step ensures that the acoustic properties of the audio are encoded in a form suitable for deep learning-based TTS systems.

To begin, raw audio waveforms are loaded from their respective .wav files using the librosa library, resampled to a standard rate of 22,050 Hz, and processed in mono to maintain consistency. The audio signal is segmented using a short-time Fourier transform (STFT) with a window size (n_fft) of 1024 samples and a hop length of 256, ensuring a sufficient temporal resolution for speech modelling.

The resulting linear spectrogram is then mapped to the perceptually motivated mel scale using a filter bank consisting of 80 mel bands, bounded by a minimum frequency of 80 Hz and a maximum of 7600 Hz. This transformation captures frequency content in a manner aligned with human auditory perception. Finally, logarithmic scaling is applied via decibel conversion (power_to_db) to compress dynamic range and enhance the numerical stability of downstream learning models.

The computed mel spectrogram is serialized and saved in NumPy format (.npy) to enable efficient loading and batch processing during training. This representation preserves the essential prosodic features while discarding phase information, which is typically reconstructed in the vocoder stage of TTS pipelines.

Step 7: Dataset Consolidation and Export

To support supervised learning, all extracted data—audio, mel spectrograms, transcripts, and emotional annotations—are collated into a unified dataset schema. This final stage generates a master CSV index that links each processed sample to its associated metadata and features, facilitating streamlined access during model training and evaluation.

The pipeline iterates over directories containing n-gram and phrase-based audio-text pairs. For each .wav file, a corresponding .csv annotation file is parsed to extract relevant emotional dimensions, including acoustic and semantic valence, arousal, speech rate, intensity (in decibels), polarity, articulation clarity, jitter, shimmer, and narrative variation. Each of these features is represented as a floating-point value aligned with a specific dimension in the N-dimensional emotion space defined by the Bibliotec system.

Subsequently, the associated mel spectrogram is computed (Step 6) and saved to a designated directory. The master CSV file is updated with a new entry containing:

- A unique identifier
- File paths to the raw audio and its corresponding mel spectrogram
- The aligned text transcript
- All computed emotional attributes
- This structured dataset facilitates downstream integration with emotion-conditioned TTS models. Each row in the master CSV represents a self-contained training instance, ensuring modularity and reproducibility across both development and deployment phases.

4.5 TTS Model Training Approaches

To actualize the vision of emotionally expressive narration in Bibliotec, two complementary approaches are being explored for developing a text-to-speech (TTS) engine capable of conditioning output on N-dimensional emotional vectors. The first approach involves fine-tuning an existing, high-performance TTS model, while the second proposes building a custom model architecture from the ground up. Both methods aim to synthesize high-quality speech that faithfully reflects the emotional and narrative context of the source text.

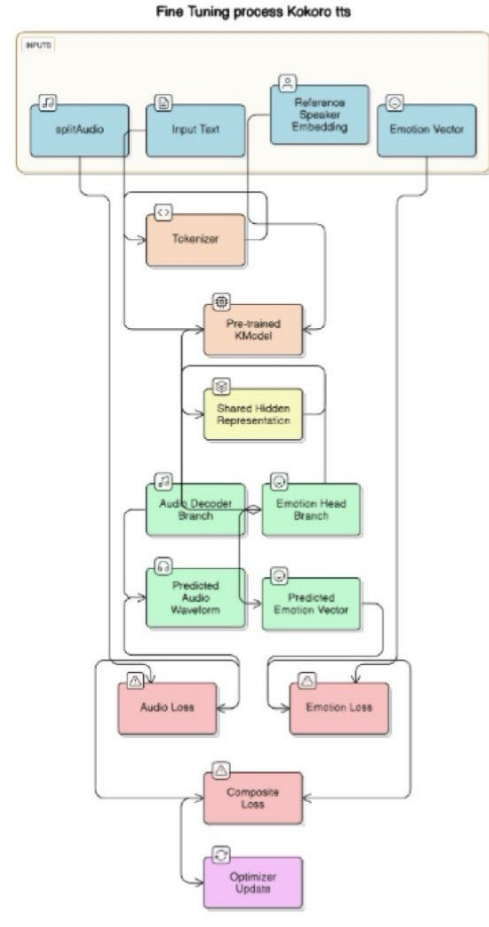


Fig 4.5: Kokoro TTS Fine-Tuning Architecture. This diagram illustrates the process for fine-tuning the pre-trained Kokoro TTS model with emotional conditioning. Input text is tokenized alongside reference audio, speaker embeddings, and emotion vectors. A shared hidden representation feeds parallel audio decoder and emotion head branches. Their respective prediction errors contribute to a composite loss, driving optimizer updates specifically for the emotion head.

Approach I: Fine-Tuning Kokoro TTS with Emotion Vectors

The initial approach leverages Kokoro TTS—an expressive, multi-task speech synthesis model renowned for its architectural flexibility and capacity to capture nuanced prosodic variations (Hexgrad, n.d. ; Communeify, 2025). This method focuses on integrating emotion conditioning into the model by fine-tuning a dedicated emotion prediction component (the emotion_head) (Lee & Tashev, 2020). The fine-tuning is performed on a curated dataset composed of phrase-level utterances, each synchronized with a 10-dimensional emotion vector derived from both textual and acoustic analyses (as described in Steps 4–5).

The training process begins with a custom dataset loader that reads a master CSV file containing references to audio waveforms, precomputed mel spectrograms, transcripts, and corresponding emotion vectors. Each training sample is structured to include:

- Spoken Phrase: The transcript provided under the header words_spoke
- Normalized Audio Waveform: The audio data loaded from the associated .wav file and normalized to mitigate amplitude discrepancies.
- Emotion Vector: A 10-dimensional vector that encodes parameters such as acoustic valence, semantic valence, arousal, speech rate, intensity (in decibels), polarity, articulation clarity, tension (jitter and shimmer), and narrative variation.
- Speaker Reference Embedding: A placeholder tensor (e.g., a zero tensor of fixed dimensions) that may be used for future speaker conditioning.

During training, the model concurrently predicts the audio waveform and the emotion vector. A composite loss function is employed, consisting of:

1. L1 Loss: Evaluating the absolute difference between the predicted and ground truth audio waveforms.
2. Mel Spectrogram Loss: Assessing perceptual fidelity by comparing the predicted and target mel spectrograms.
3. Mean Squared Error (MSE) Loss: Quantifying the error in the predicted emotion vector relative to the ground truth.

To ensure stable convergence, only the parameters within the *emotion_head* are unfrozen and updated during fine-tuning; the remainder of the model's pre-trained weights are preserved. Additionally, the weight applied to the emotion loss is initially down-weighted and then progressively increased over the early epochs, facilitating a gradual and smooth integration of affective features into the synthesis process.

A cosine annealing learning rate scheduler and gradient clipping are utilized to maintain training stability and prevent gradient explosion. Upon completion of training, the final model state is saved, making it ready for deployment in inference scenarios or further iterative refinement. This approach effectively exploits existing speech synthesis capabilities while enhancing the expressiveness of the output through fine-grained emotional conditioning.

Approach II: End-to-End TTS with Integrated Text and Emotion Conditioning

In this approach, we develop an end-to-end text-to-speech (TTS) synthesis system that learns to generate speech exclusively from a textual input paired with a 10-dimensional emotion vector (Wang et al., 2017 1 ; Jung & Lee, 2023 2). Unlike conventional methods that require multiple modalities as input, our pipeline leverages emotion solely as a conditioning signal, while treating audio—specifically, mel spectrograms and raw waveforms—as the target output during training.

The data ingestion and preprocessing stage begins with constructing a dataset from a master CSV file. Each sample in this dataset includes a transcript (found under the header *words_spoke*), a precomputed mel spectrogram stored as a NumPy (.npy) file, the corresponding raw audio waveform from a .wav file, and a 10-dimensional emotion vector. The emotion vector encodes various acoustic and linguistic features such as acoustic and semantic valence, arousal, speech rate, intensity in decibels, polarity, clarity of articulation, as well as measures of tension like jitter and shimmer, and narrative variation. A vocabulary is built from the transcripts and saved as a file named *vocab.txt*. A helper function loads this vocabulary to tokenize the transcripts into sequences of integer token IDs. The custom dataset loader (implemented in the SpeechDataset class) then packages each sample with the mel spectrogram (serving as the decoder target), the tokenized text, the emotion vector, and the waveform for final loss computation.

The model architecture is composed of three primary modules, each implemented as a PyTorch submodule. The first component is the main encoder, which fuses features from both text and emotion inputs. The text encoder within this module starts with an embedding layer (Hinton et al., 2012) that converts token IDs into dense embeddings, using the vocabulary loaded from vocab.txt. A GRU (Cho et al., 2014) processes these embeddings to capture sequential dependencies, and the final hidden state is projected through a linear layer to produce a latent text representation. In parallel, an emotion encoder—a simple feedforward network (Rumelhart et al., 1986) with ReLU activations—transforms the 10-dimensional emotion vector into an emotion latent representation. These two latent representations are then concatenated and passed through an additional linear layer with a non-linear activation to yield a unified latent vector.

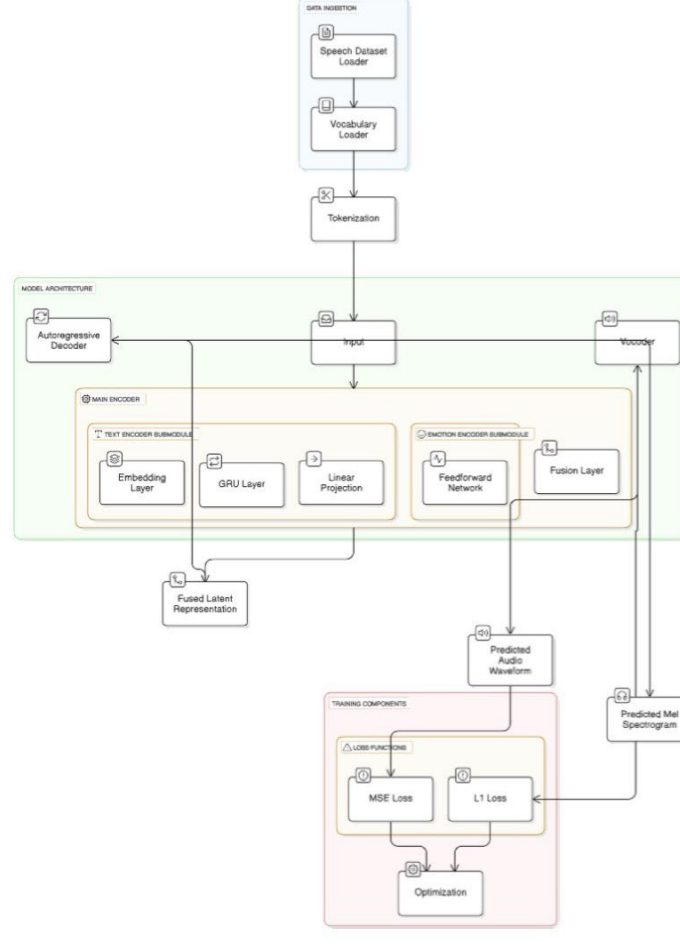


Fig 4.5.1: End-to-End Emotion-Conditioned TTS Model. This schematic outlines the custom TTS pipeline. Data ingestion involves loading datasets and tokenizing text. The model architecture features a main encoder with text (embedding, GRU) and emotion (feedforward network) submodules, producing a fused latent representation. This drives an autoregressive decoder and a vocoder to generate audio. L1 and MSE losses guide the training optimization process.

This unified latent representation is fed into an autoregressive decoder, which generates a mel spectrogram frame-by-frame. At each time step, the decoder receives both the fused latent representation and the previous mel frame (with teacher forcing applied during training), and uses a prenet composed of several linear layers with dropout and ReLU activations alongside a GRU to predict the next mel frame. A final linear projection converts the GRU's hidden state to match the mel frequency dimension. Finally, the vocoder takes the predicted mel spectrogram and converts it into a raw audio waveform using a series of transposed convolutional layers. The final output is passed through a tanh activation to ensure the waveform values lie within the range [−1,1.

The training pipeline is built around a composite loss function that encourages both spectral and waveform fidelity. An L1 loss is used to measure the absolute differences between the predicted and target mel spectrograms, while a mean squared error (MSE) loss quantifies the discrepancy between the synthesized and actual audio waveforms. To maintain training stability, techniques such as gradient clipping and teacher forcing are employed. The training loop, implemented in a Python script (train_tts.py), loads the vocabulary from *./self_model/encoders/vocab.txt*, creates the custom Speech Dataset (integrating tokenization), and initializes the model with the correct vocabulary size. Mini-batches are processed iteratively, with forward passes, loss computation, gradient backpropagation, and parameter updates, while periodic checkpointing of the model state allows for later inference.
After training, the model is evaluated using an inference script (test_model.py). This script loads the saved model checkpoint, tokenizes a sample sentence (for example, "the quick brown fox") using the same vocabulary, and then generates a mel spectrogram and the corresponding waveform from the fused latent representation. The output waveform is then saved as a .wav file for auditory evaluation.
Overall, this integrated approach demonstrates the potential to synthesize expressive speech by conditioning solely on text and emotion. Although the audio is not used as an input, it is critical as a target during training to guide the model in generating accurate mel spectrograms and waveforms. This end-to-end architecture, with its flexible encoder-decoder-vocoder pipeline, lays a robust foundation for further improvements, such as incorporating more advanced vocoders, additional loss functions, or even extra conditioning modalities, to ultimately produce more natural and expressive speech.

5. EVALUATION

In the first approach, I fine-tuned a pre-trained TTS model—Kokoro TTS—with an added emotion conditioning module. Across several experimental runs using different datasets, the best-performing configuration managed to produce speech that was around 60% intelligible. While that figure might sound modest, it marked a significant step: the model successfully embedded emotional nuance into the narration without seriously degrading overall audio quality. The benefit of starting with a robust architecture like Kokoro became clear early on—it converged faster, was less prone to instability, and didn't require an enormous dataset to start producing usable results. That said, the system still struggled to fully convey expressiveness in some instances, which suggests that clearer emotional delivery might require either more fine-tuning or a richer, better-annotated dataset.

The second approach was more ambitious: I built a custom end-to-end TTS pipeline from scratch, designed to generate both mel spectrograms and waveforms from nothing more than raw text and a 10-dimensional emotion vector. Structurally, the pipeline included a main encoder (handling both text and emotion), an autoregressive decoder, and a vocoder. On paper, the design was sound—and in practice, it did manage to generate audio. But the quality was clearly at an early, experimental stage. The output was sometimes garbled, and although it resembled speech, it lacked the clarity and naturalness of more mature systems. Training it was also a steep hill: long sessions, significant GPU use, and a fairly large amount of trial and error. Still, the fact that this homegrown model could produce sound at all—without relying on pre-trained audio encoders—is an encouraging sign. It shows there's a viable foundation here, even if the current version is just a starting point. There's definite potential for improvement with architectural tweaks, refined loss functions, and more targeted training strategies

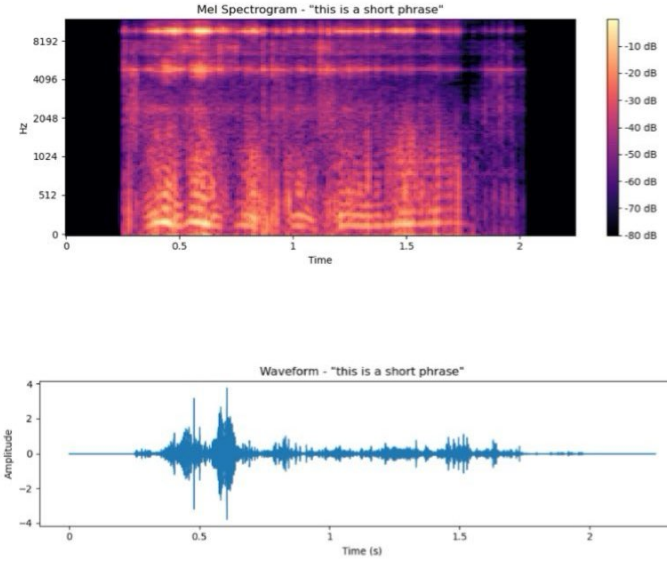
Ultimately, these two approaches reflect a familiar trade-off in speech synthesis. Fine-tuning an existing model is faster, more reliable, and gets you something useful more quickly. But if you're chasing creative control or aiming to push the boundaries of what your TTS system can express, building from scratch—though costlier—offers unmatched flexibility. In the context of Bibliotec, both directions taught me a lot. One gave me a practical, deployable tool. The other gave me a glimpse of what could be possible down the road.

Fig 5: Training Log for Approach 2. This console output displays the execution of train_tts.py, showing gradient norms for model layers, per-batch Mel and Wave losses, and resource usage during optimization.

```
(selfmodeltts) PS C:\Users\vhalan\Desktop\scriptbuild> python .\train_tts.py
Using device: cuda

Epoch 1/1
text_embedding.weight: grad norm = 0.0001
text_encoder.weight_ih_l0: grad norm = 0.0019
text_encoder.weight_hh_l0: grad norm = 0.0004
text_encoder.bias_ih_l0: grad norm = 0.0002
text_encoder.bias_hh_l0: grad norm = 0.0001
text_fc.weight: grad norm = 0.0018
text_fc.bias: grad norm = 0.0003
emotion_net.0.weight: grad norm = 0.5396
emotion_net.0.bias: grad norm = 0.0002
emotion_net.2.weight: grad norm = 0.8454
emotion_net.2.bias: grad norm = 0.0004
fusion.0.weight: grad norm = 2.8011
fusion.0.bias: grad norm = 0.0009
Batch 1: Mel Loss: 33.95, Wave Loss: 1.00
RAM: 23356.65 MB
GPU: 51.45 MB
...
text_embedding.weight: grad norm = 0.0000
text_encoder.weight_ih_l0: grad norm = 0.0006
text_encoder.weight_hh_l0: grad norm = 0.0001
text_encoder.bias_ih_l0: grad norm = 0.0001
text_encoder.bias_hh_l0: grad norm = 0.0000
text_fc.weight: grad norm = 0.0006
text_fc.bias: grad norm = 0.0001
emotion_net.0.weight: grad norm = 0.1591
emotion_net.0.bias: grad norm = 0.0000
emotion_net.2.weight: grad norm = 0.3074
emotion_net.2.bias: grad norm = 0.0001
fusion.0.weight: grad norm = 0.9688
fusion.0.bias: grad norm = 0.0003
Batch 11: Mel Loss: 46.65, Wave Loss: 1.01
RAM: 23357.30 MB
GPU: 51.41 MB
Training complete
```

Fig 5.1: Approach 1 Output Spectrogram. The distinct visual structures correspond to an intelligible waveform generated for "this is a short phrase"



6. IMPLEMENTATION

This section details the practical implementation of the Bibliotec pipeline. While Section 4.5 theoretically proposed advanced methodologies like fine-tuning Kokoro TTS or building an end-to-end emotion-conditioned model, these approaches proved computationally infeasible for this project. The final implementation relies instead on a pre-trained Kokoro TTS engine, simulating emotional variation via modulation of prosodic inference parameters. This decision was driven by significant resource constraints. Firstly, the sheer computational power required for training was prohibitive; initial estimates suggested that surpassing the quality of the pre-trained Kokoro TTS would necessitate 40,000-50,000 H100 GPU hours. Secondly, the process of preparing the necessary annotated dataset—including extracting features and validating the accuracy of emotion vectors—was extremely time-intensive, with preliminary work indicating approximately 4 hours of annotation effort needed for every 2 hours of audio data. Basic training trials (200 epochs with a batch size of 400) already consumed 10 hours on an RTX 4070 Super, highlighting the impracticality of scaling this for the project's scope. Therefore, the adopted method represents a pragmatic solution, serving as a successful proof-of-concept within the available resource limitations

6.1 System Architecture Overview

The implementation follows a modular, script-based architecture that processes each book sequentially based on its unique identifier. At the heart of the system is a master script (main.py) that coordinates data retrieval, cleaning, multimodal generation, and storage via subprocess execution and API calls. This framework allows for scalability and modular enhancement while ensuring reproducibility across different input texts (Goodfellow et al., 2016).

6.2 Pipeline Execution Flow

The automated pipeline reads a list of book identifiers from book_ids.txt and performs a series of standardized steps for each entry:

Step 1: Book Retrieval and Chapter Segmentation (LLM-Assisted)
In the revised implementation, textual content for each book is retrieved from Project Gutenberg using a robust downloader that supports multiple fallback URL patterns to ensure high reliability. Once downloaded, the text undergoes a two-stage transformation process leveraging a locally hosted large language model (LLM Meta-Llama-3.1-8B-Instruct.

First, the raw content is passed through a start-offset detector, which prompts the LLM to identify and return the character index at which the main narrative begins. This offset is used to remove all introductory content, including licensing statements, acknowledgments, author metadata, and any front matter such as tables of contents or prefaces. This ensures that only narratively relevant material is retained for further processing.

Second, the cleaned text is analysed using a separate LLM prompt designed to infer the structure of chapter headings. The model evaluates the formatting patterns such as "CHAPTER I", "BOOK TWO: CHAPTER I", or Roman numeral-only lines—and returns a structured description of the dominant format. This description is then mapped to a regular expression used to segment the narrative into discrete chapters.

If the LLM fails to produce a confident result, a set of heuristic fallback patterns are applied to recover structural divisions. All extracted chapters undergo further filtering to ensure they meet a minimum content threshold, and valid segments are written to individual text files for downstream audio synthesis and alignment. Metadata and processing status are concurrently updated in the system's tracking CSV.

This LLM-enhanced pipeline—anchored by Meta-Llama-3.1-8B-Instruct—ensures both higher segmentation accuracy and greater adaptability across diverse literary formats, resolving prior issues of misaligned chapters and noisy introductory matter.

Step 2: Preprocessing and Cleaning
Using clean_chunks.py, each textual segment is sanitized to remove special characters, non-verbal annotations, and other artifacts that may degrade TTS performance. This ensures compatibility with the downstream speech synthesis engine.

Step 3: Cinematic Cover Prompt and Image Generation
A dynamic prompt is generated for each book cover using the OpenAI GPT-3.5 API. The prompt is formulated based on the book's title, author, and genre(s), and follows a template designed to emphasize visual drama, widescreen layout, and thematic richness. The generated prompt is then passed to DALL·E 3, which produces a high-resolution (1792×1024) image saved locally as cover.png.

In cases where the metadata file (info.txt) is missing or incomplete, a generic fallback prompt is employed.

Step 4: Audio Generation via Pre-Trained Kokoro TTS
Unlike the fine-tuning and end-to-end training pipelines described theoretically in Section 4.5, the implemented system uses a pre-trained Kokoro TTS model in inference-only mode. Emotional expressiveness is achieved through controlled manipulation of Kokoro's inference-time parameters—specifically pitch, duration, energy, and intonation contours—modulated using the extracted emotion vectors.

This decision was informed by empirical observations: the dataset assembled for fine-tuning (with less than 30 hours of high-quality, emotion-annotated narration) proved insufficient to support meaningful learning, while the training process for an end-to-end model required compute and data resources beyond the scope of this implementation. Furthermore, early trials of model training yielded intelligibility rates below acceptable thresholds, reaffirming the decision to adopt inference-based expressiveness instead.

Step 5: Subtitle Synchronization
Each audio chunk is paired with a corresponding subtitle track using generate_srt_batch.py, which aligns spoken phrases with time-coded subtitles in .srt format. These files provide accessibility support and facilitate the integration of visual feedback in future UI components.

Step 6: Compression of Media Files
To optimize storage and cloud delivery, the generated media—including .mp3 audio files and .png cover images—is compressed using compress_audio_and_image.py. This step preserves perceptual quality while minimizing file size.

Step 7: Deployment to Azure Cloud
The final step involves the upload of all media and metadata to Azure Blob Storage, organized by book ID. Each container includes the following files:

- cover.png (cinematic image),
- .mp3 audio chunks,
- .srt subtitles,
- info.json (metadata manifest).

This cloud-first design facilitates scalability and cross-platform deployment, enabling mobile and web access to the synchronized, expressive narration experience.

6.3 Limitations and Future Expansion

The implemented system successfully generates expressive TTS by employing heuristic control of prosodic parameters on a pre-trained model, foregoing fine-tuning due to practical constraints on dataset size and compute time. This approach proved to be the most pragmatic and effective strategy under the circumstances, yielding impressive results. It achieves remarkable efficiency, generating approximately 10 hours of high-quality audio narration within just one hour of processing time. Consequently, the resulting TTS output reaches a very good level of quality and expressiveness, effectively meeting the project's core goals despite not implementing the originally proposed fine-tuning or end-to-end training methodologies. While future iterations with expanded datasets and distributed training could potentially unlock even greater emotional fidelity through fine-tuning or hybrid models, the current method represents a highly successful and resource-efficient solution.

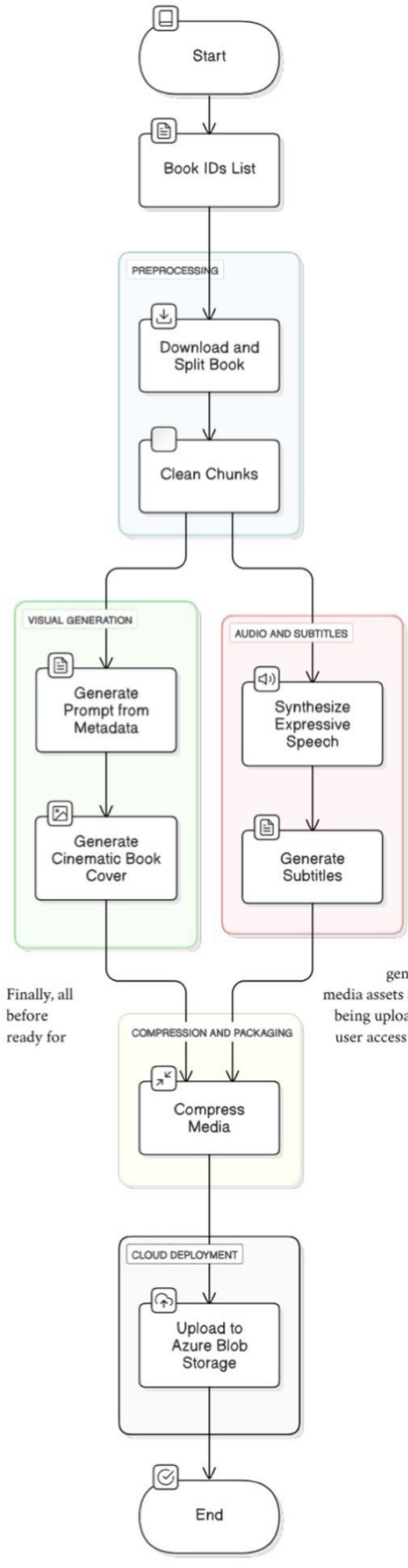


Fig 6: Bibliotec Automated Content Pipeline. This flowchart illustrates the end-to-end process for ingesting and preparing books. Starting with a list of book IDs, the pipeline downloads and preprocesses the text, cleaning and segmenting it. Parallel branches then handle visual and audio generation: one creates cinematic cover art from metadata prompts, while the other synthesizes expressive TTS audio and generates corresponding subtitles. Finally, all media assets are compressed and packaged before being uploaded to Azure Blob Storage, ready for user access within the Bibliotec platform.

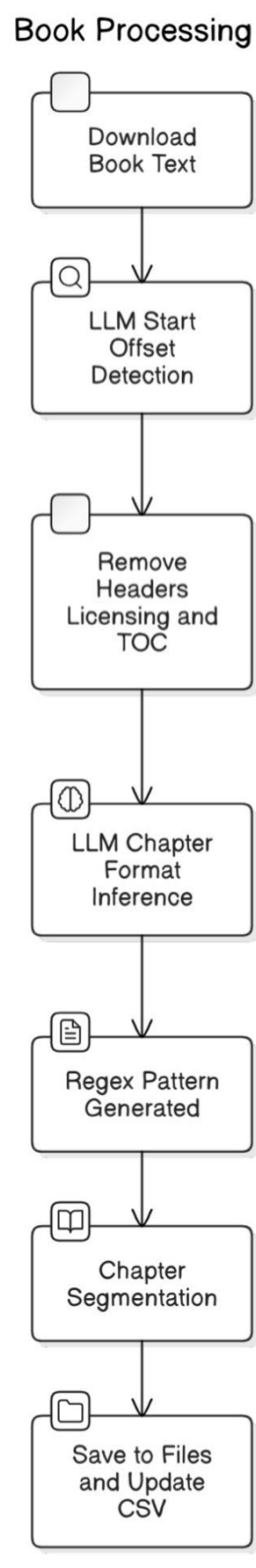


Fig 6b: LLM-Enhanced Book Preprocessing and Segmentation. This flowchart shows the initial text processing stages. After downloading, an LLM identifies the content start offset, allowing removal of headers and licenses. Another LLM infers the chapter format, generating a regex pattern for precise chapter segmentation. Finally, segmented chapters are saved as separate files for downstream processing.

7: SEARCH

The search functionality is a cornerstone of the Bibliotec platform, designed to provide users with flexible and efficient ways to discover literary works within the digital library. Implemented through the AdvancedSearchPage component, the feature prioritizes a responsive user experience, leveraging client-side filtering powered by a pre-compiled index fetched from cloud storage.

7.1 User Interface and Interaction

The search interface presents users with two primary methods for finding books:

Top Search Bar: A prominent, always-visible search input allows users to quickly search across book titles and authors to enhance performance and user experience, input changes are debounced using a DEBOUNCE_DELAY of 300 milliseconds. This ensures that the filtering logic (filterBooks) is only triggered after the user pauses typing, preventing excessive re-renders during input. Visual feedback, including a loading spinner (spinner) within the search icon wrapper, indicates when a search operation is in progress (isSearching state).

Advanced Search Filters: Users can access more granular search options by toggling the "Advanced Search" section. This expandable area (advsearch-dropdown) contains specific input fields for:

Keyword (searching title/author)
Specific Title
Specific Author
Year These fields are managed using React refs (keywordRef, titleRef, authorRef, yearRef). Users can apply these filters explicitly using the "Apply Filters" button or reset them via the "Reset Filters" button, both triggering the filterBooks logic.
Accessibility is considered through the use of ARIA attributes (aria-expanded, aria-controls, aria-hidden, aria-label, aria-live="polite") to ensure the interface is navigable and understandable for users relying on assistive technologies.

7.2 Data Indexing and Retrieval

The search functionality relies on a static index file, books_index. json, which is generated offline by the Python script generate_books_index.py. This script performs the following steps:

Connects to Azure Blob Storage: Uses the AZURE_BLOB_CONNECTION_STRING.
Scans Containers: Iterates through containers in the storage account (where each container typically represents a single book identified by its name/ID).
Extracts Metadata: For each container, it searches for specific blobs:
cover.png: Retrieves the public URL for the book cover.
info.json: Downloads this file, parses it to extract the book's title, author, and year. It also calculates the total book length by summing the duration fields within the chapters array found in info.json, formatting the result as "Xh Ym".
Compiles Index: Aggregates the extracted data for all books into a single dictionary (all_books), where keys are the container names (book IDs).
Uploads Index: Writes the dictionary to a local books_index.json file and then uploads it to a designated Azure Blob Storage container (indexes) with the appropriate content type (application/json).

7.3 Client-Side Filtering

Upon loading the AdvancedSearchPage, the React component fetches the books_index.json from its Azure Blob Storage URL (https://audiobookinformation.blob.core.windows.net/indexes/books_index.json). This entire index (booksIndex state) is stored client-side.

The filterBooks function performs the search logic directly on this local data. It converts all search terms and book metadata fields (title, author, year) to lowercase for case-insensitive matching and uses .includes() to check for partial matches based on the active topQuery and any applied advanced filters. This client-side approach ensures that subsequent searches and filtering adjustments are fast and responsive, as they don't require further network requests after the initial index load.

7.4 Results Display

Filtered results (searchResults state) are displayed in a responsive grid (results-grid). Each book is rendered as a ResultCard component:

It uses the next/image component for optimized loading of the cover_url, displaying a placeholder (card-image-placeholder) if the cover is missing. Responsive image sizes are specified for different viewports.
Displays the book's title, author, year, and calculated length.
Each card is wrapped in a next/link component, directing the user to the /webaudioreader page with the corresponding book ID passed as a query parameter (?bookName=${book.id}).
User feedback messages inform the user about the status (e.g., "Loading books...", "Searching...", "No results found...").

The design and layout of a website—its name, logo, and overall aesthetic—are crucial not only for capturing user attention but also for retaining it. Beyond making a strong first impression, the design must facilitate deeper engagement to foster brand trust, preference, and repeat interactions. If visitors merely glance at the homepage or landing page without exploring further or using its services, the design fails its core objective of converting visits into sustained engagement.

A well-thought-out user experience (UX) starts with an interface that is both visually appealing and immediately understandable. Key design principles include:

Intuitive Layout and Navigation
A homepage should clearly present the brand name and logo alongside a succinct value proposition. This immediate clarity allows users to grasp the site's purpose at a glance, encouraging further exploration. Logical menu structures with clear, concise navigation links help reduce cognitive load and prevent confusion, enabling users to easily locate the information they need without being overwhelmed by excessive options.

Unobstructed Experience
An effective design minimizes visual clutter by avoiding excessive pop-ups, busy backgrounds, and competing elements that detract from the core content. A consistent design language—uniform fonts, colour schemes, and button styles—creates predictable interactive patterns, allowing users to learn and navigate the interface effortlessly.

Accessibility for Low-Resource Devices
High-resolution images and complex animations, while attractive, can impede performance on older devices or slow networks. By optimizing media through image compression and using lightweight design elements, websites can remain responsive and functional. Additionally, offering a simplified version of the homepage—omitting heavy 3D elements or advanced scripts—ensures that essential functionalities like browsing, searching, and reading are accessible to all users. Progressive enhancement further allows advanced features to be layered on for devices that can support them.

Encouraging Deeper Engagement
Design elements such as clear calls to action (CTAs) guide users toward important features—for example, "Explore the Library" or "Start Listening." Purposeful interactive components (like a 3D model demonstrating AI narration) add real value rather than serving as mere decoration. Furthermore, a frictionless sign-up or onboarding process is vital for transforming passive viewers into active participants.

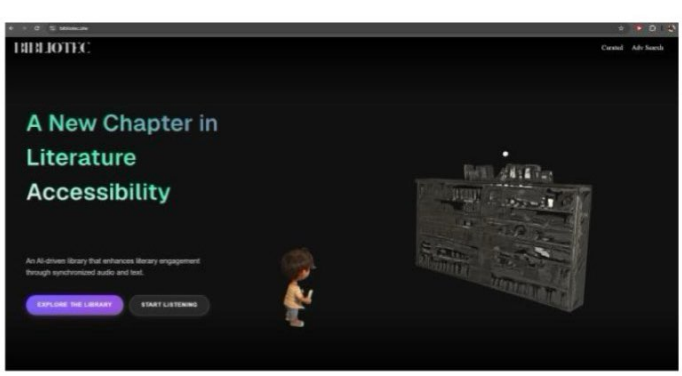


Fig 8.2 : Bibliotec Landing Page. Modern dark-mode design emphasizing literature accessibility with engaging visuals and clear calls-to-action.

Building Brand Trust and Preference
Consistent messaging that reinforces the brand's values—such as accessibility, innovation, and ethical practices—builds trust. Showcasing user testimonials and social proof can further encourage new visitors to engage deeply and develop long-term loyalty.

By balancing immersive visual elements with performance-optimized simplicity, web designers can create platforms that not only capture attention but also convert visits into meaningful, sustained engagement

Design. BIBLIOTEC

fig 8.1: The logo for the platform is typographically based on its name. Bibliotec is derived from the French word bibliothèque, as well as the Latin bibliotheca, from which many languages take their word for library. The addition of tec implies a connection to technology, thus conveying the idea of a technological library. The use of a sans-serif font reflects a modern and minimalist aesthetic.

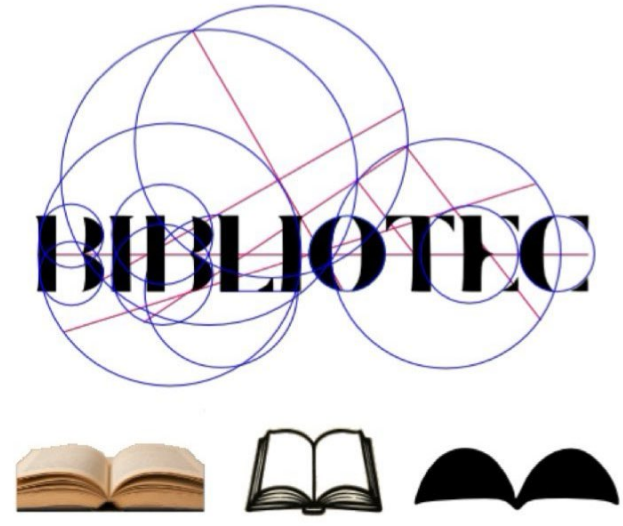


Fig 8.1.1: presents a geometrical breakdown of the Bibliotec logo, showcasing the precision and intentionality behind its design. The blue circles and magenta lines illustrate the use of consistent radii, spacing, and alignment principles to construct each letterform. This methodology reflects the platform's core values: structure, clarity, and modernity. The integration of circular geometry not only enhances visual balance but also evokes a sense of cohesion and harmony across the logotype. The logo draws inspiration from the traditional form of a book. The open book shape is subtly mirrored in the letterforms, particularly in the curvature and openness of letters like "b" and "e".

8.2 Website Design

In alignment with Bibliotec's commitment to accessibility and inclusive design, the platform adopts a dual landing page strategy. The primary landing page is optimised for user engagement and conversion, employing rich media, interactive components, and persuasive content to effectively communicate the platform's value proposition. Complementing this is a lightweight alternative, specifically designed for low-resource devices and constrained network environments. This page minimises visual and computational load, ensuring functional accessibility across a broad spectrum of devices. Such an approach supports Bibliotec's objective of technological inclusivity, ensuring that users—regardless of device capability—can engage with the platform's digital library infrastructure.

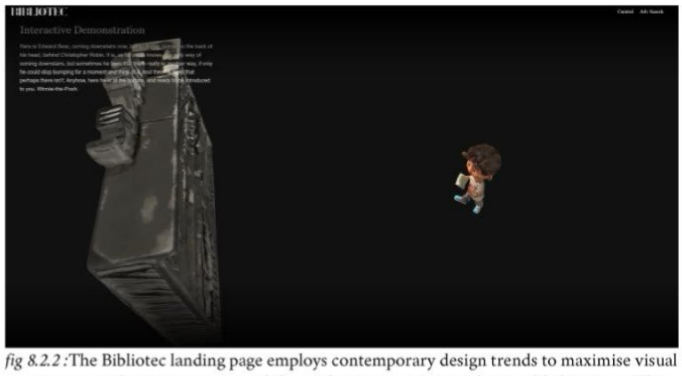
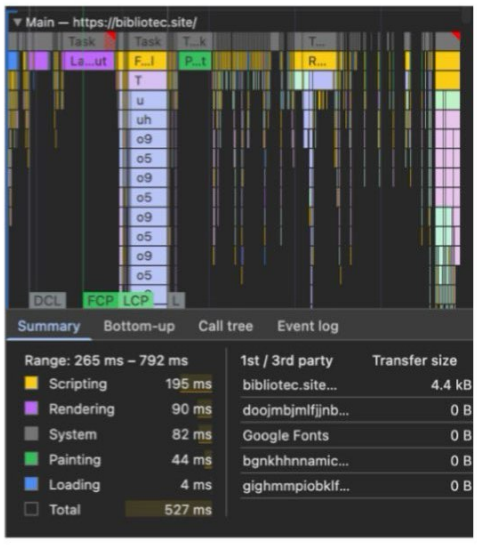


*fig 8.2.2 :*The Bibliotec landing page employs contemporary design trends to maximise visual engagement and user conversion, while reinforcing its mission of accessible literature. The use of a dark mode aesthetic, gradient typography, and 3D elements creates a modern, immersive interface that draws attention and enhances readability. Minimalist layout and clear call-to-action buttons reduce cognitive load and guide users seamlessly toward key interactions. Emotive messaging and a concise value proposition establish trust and clarity within seconds. The inclusion of a child figure introduces warmth and inclusivity, aligning with Bibliotec's focus on accessibility across age groups and devices, ultimately supporting its conversion goals.

fig 8.2.3: The performance trace of the Bibliotec landing page reveals strong frontend optimisation, with a Largest Contentful Paint (LCP) recorded at 0.53 seconds, well within recommended thresholds for perceived load speed. The Cumulative Layout Shift (CLS) remains at zero, indicating a visually stable experience during load.

Fig 8.2.4: Breakdown of task timings shows a total load time of 527 milliseconds, with the bulk of activity concentrated in scripting (195 ms) and rendering (90 ms), followed by system operations (82 ms) and painting (44 ms). Notably, the actual network transfer size remains minimal, with core assets such as fonts and scripts contributing little to page weight. Despite rich visual elements, the efficient scheduling of rendering and layout tasks ensures that critical visual content becomes interactively available without delay.



8.3 Web map / Database

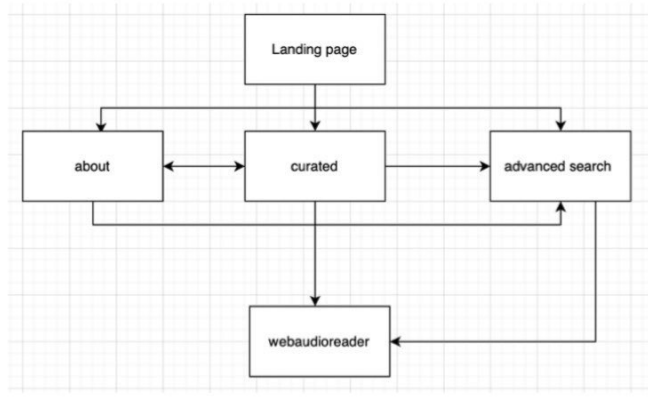


Fig 8.3: The sitemap highlights the user-friendly navigation of the Bibliotec platform. From a central landing page, users can easily access curated content, learn more via the about page, perform advanced searches, or launch the web audio reader. Each section is interconnected, ensuring smooth transitions and minimal friction, enhancing overall accessibility and ease of use

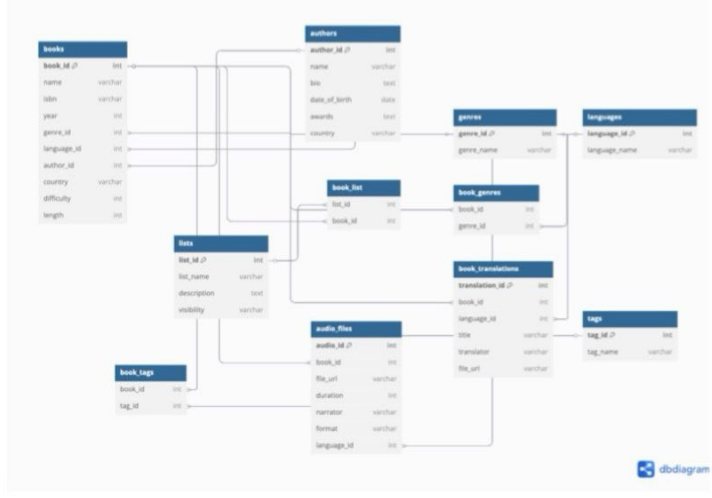


Fig 8.3.1: This section outlines the theoretical relational schema initially envisioned for Bibliotec, designed to provide a robust and scalable foundation. Conceptually, a central books table holding essential metadata (title, author, etc.) would link to related tables like authors, genres, and languages. Join tables (book_genres, book_tags) were planned for flexible categorization, while audio_files and book_translations tables would manage narrated versions and different language editions, respectively. User curation features were supported via lists and book_list tables.

This idealized relational structure was designed to offer efficient querying, ensure data consistency, and support complex relationships inherent in a large digital library. However, in practice, a different, more resource-saving approach was implemented to align with project scope and development constraints. Instead of deploying a full relational database, the final system primarily utilizes flat files (like info.json) stored alongside media assets within individual book containers in cloud storage (Azure Blob Storage). While sacrificing complex querying capabilities, this simplified the deployment architecture and allowed resources to be focused on the core audio synthesis and synchronization features

9: FUNCTIONALITY

9.1 Overview

The Bibliotec web platform (bibliotec.site) serves as the primary user interface for accessing the library of AI-narrated literary works. Developed using a React/Next.js frontend and leveraging Azure Blob Storage for content delivery, the site is designed to provide an engaging, accessible, and discoverable reading experience. Key functionalities include Browse curated collections, performing advanced searches, and interacting with the core synchronized audio-text reader. An effective design minimizes visual clutter by avoiding excessive pop-ups, busy backgrounds, and competing elements that detract from the core content. A consistent design language—uniform fonts, colour schemes, and button styles—creates predictable interactive patterns, allowing users to learn and navigate the interface effortlessly.

9.2 User Interface and Navigation

The platform presents a clean, modern interface, typically employing a dark-mode aesthetic for comfortable reading (as seen in Fig 3.1 and Fig 9.X [referencing the screenshot you provided]).

Landing Page: The initial entry point introduces Bibliotec's mission of enhancing literature accessibility, featuring key visuals and clear call-to-action buttons like "Explore the Library" or "Start Listening" to guide users toward content.

Main Navigation: A persistent header contains the Bibliotec logo and primary navigation links, typically including "Curated" for Browse pre-defined collections and "Adv Search" for targeted searching.

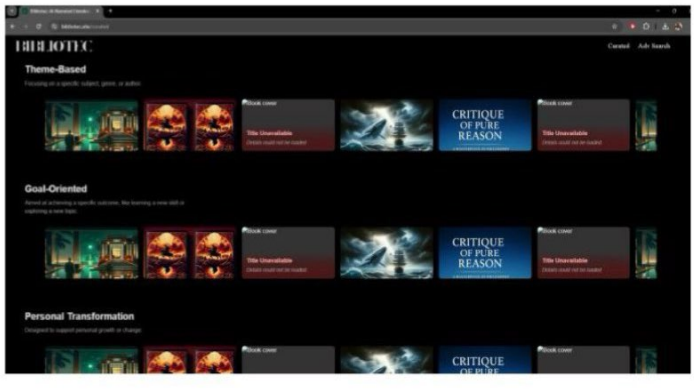


Fig 9.2 : Thematic rows display book cards with covers, facilitating visual Browse and exploration.

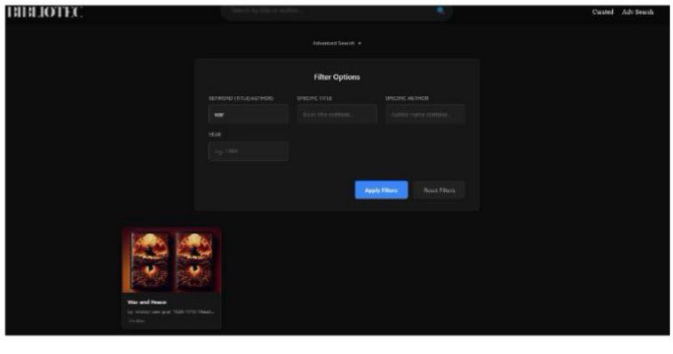


Fig 9.2.2 : Advanced Search Interface. Shows filter options including keyword, specific title, author, and year. Users can apply or reset these filters to refine search results displayed below.

9.3 Content Discovery

Bibliotec facilitates book discovery through two main mechanisms:
Curated Collections (Fig 9.X [referencing the screenshot]): This view organizes books into thematic categories (e.g., "Theme-Based", "Goal-Oriented", "Personal Transformation"). Each category displays relevant books as visual cards, showing generated cover art and titles (or placeholders if data is unavailable). This allows users to browse based on mood, interest, or learning goals.
Advanced Search: Accessed via the "Adv Search" link, this feature allows users to find specific books using various criteria. As detailed in Section 7, users can search by general keyword (title/author) or filter by specific Title, Author, or Year. The search operates efficiently using a client-side index, displaying results as a grid of book cards that link directly to the reader interface.

9.4 The Reading Experience

The core functionality is delivered through the web audio reader interface (typically accessed via /webaudioreader):

Synchronized Audio-Text Playback: This is Bibliotec's central feature. As the AI-generated audio narration plays, the corresponding text passage is highlighted on screen in real-time. This dual-modal presentation aims to improve engagement and comprehension.
Expressive Narration: The audio is generated using the modified Kokoro TTS engine (detailed in Section 6), aiming for more natural and expressive delivery compared to standard TTS.
Subtitle Display: Synchronized subtitles (.srt files) are available for each audio track, enhancing accessibility and offering an alternative mode of engagement.
Playback Controls: Users are provided with standard controls for managing the audio playback, such as play/pause, volume adjustment, and potentially seeking within the track or navigating between chapters/sections.

Fig 9.4 : Bibliotec Web Audio Player Interface. Displays the current text passage synchronized with audio narration. Features include chapter navigation via a sidebar, standard playback controls (play/pause, volume, progress bar), and a toggle for subtitles, enabling the core multimodal reading experience

9.5 Responsiveness and Accessibility

The website is built with responsive design principles to ensure a consistent and usable experience across different devices (desktops, tablets, mobiles). Accessibility features include the provision of subtitles and the use of appropriate ARIA attributes in the search interface (Section 7.1) to support users with assistive technologies.

Fig 9.5 : Bibliotec Landing Page Mobile View. Demonstrates responsive design adapting layout, visuals, and navigation for smaller screens, ensuring accessibility and preserving core messaging.

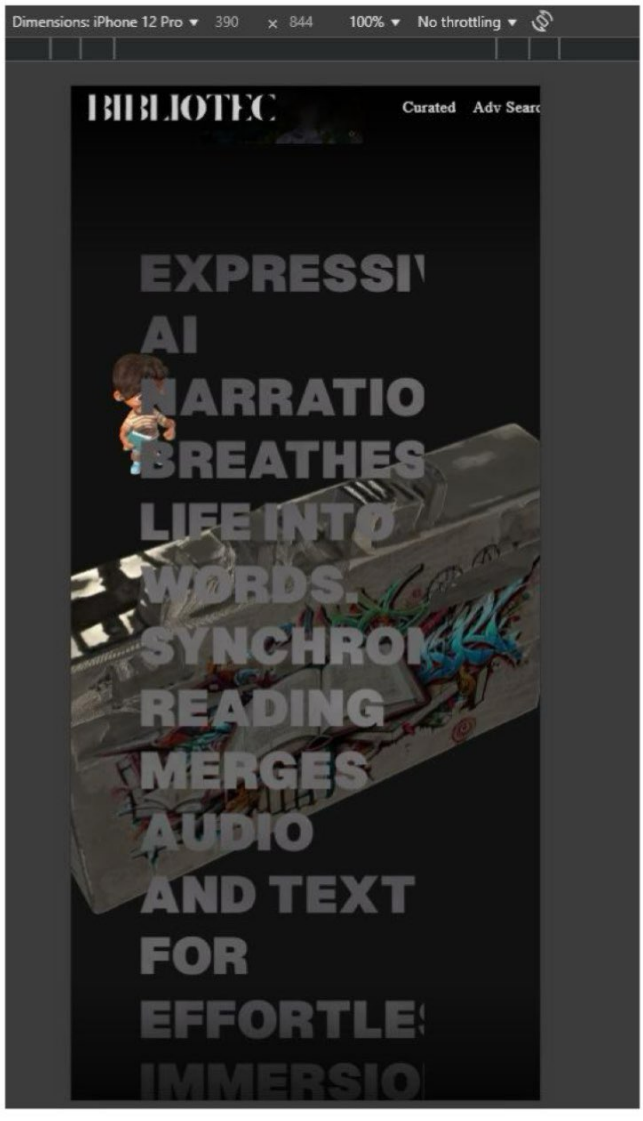
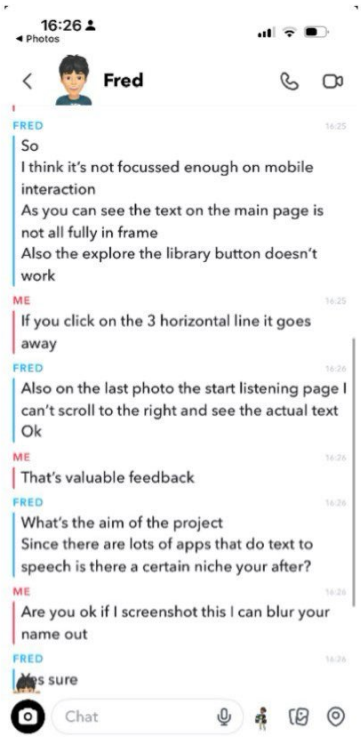


Fig 9.5.1: Shows responsive adaptation with a collapsed navigation menu, central text display for synchronized reading, and simplified audio playback controls (previous, play/pause, next) optimized for touch interaction on smaller screens.

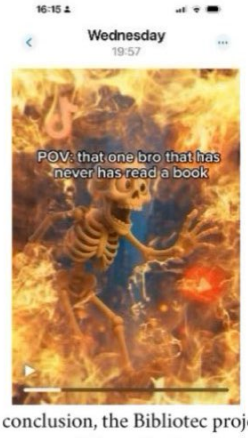
SECTION 10: USER TESTING

To evaluate Bibliotec's usability and core features, user testing was conducted with participants from diverse backgrounds, ensuring ethical procedures like informed consent were followed. General feedback was overwhelmingly positive, confirming the platform's potential beyond its initial scope. Users frequently highlighted the intuitive design and the engaging nature of the synchronized audio-text experience, noting its effectiveness in enhancing focus and immersion. Constructive feedback identified specific areas for improvement, particularly mobile usability issues such as text framing, button functionality, and scrolling limitations within the reader experience.

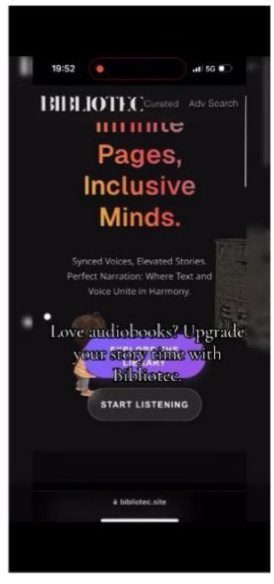

16:26
Photos
Fred
FRED
So
I think it's not focussed enough on mobile interaction
As you can see the text on the main page is not all fully in frame
Also the explore the library button doesn't work
ME
If you click on the 3 horizontal line it goes away
FRED
Also on the last photo the start listening page I can't scroll to the right and see the actual text
Ok
ME
That's valuable feedback
FRED
What's the aim of the project
Since there are lots of apps that do text to speech is there a certain niche your after?
ME
Are you ok if I screenshot this I can blur your name out
FRED
s sure
Chat

While the primary focus of this project was technical development and evaluation, the resulting Bibliotec platform possesses features suitable for targeted marketing strategies. The core audience likely includes individuals facing challenges with traditional reading engagement due to shortened attention spans shaped by digital media, alongside users seeking more immersive and accessible literary experiences. Marketing could effectively highlight Bibliotec's unique value proposition: the synchronized audio-text feature combined with expressive AI narration, designed to enhance focus, comprehension, and overall reading enjoyment.

Potential marketing channels could leverage digital platforms frequented by the target demographic. Social media like TikTok, Instagram, or YouTube could be used to showcase the platform's engaging dual-modal experience through short, dynamic video demonstrations. Content might address the common issue of declining reading habits, potentially using relatable humor or highlighting the app's ability to maintain focus in a distracting world. Outreach to educational communities, book clubs, and accessibility forums could further raise awareness. Emphasizing the platform's modern, intuitive design, positive user feedback regarding engagement, and its use of accessible public domain works would be key messaging points across all promotional efforts.



In conclusion, the Bibliotec project serves as a successful proof-of-concept. It not only met its primary objectives by delivering a functional and personally effective tool but also demonstrated significant potential for wider user engagement and accessibility, as confirmed by positive user testing. It showcases the practical application of TTS and web technologies while realistically acknowledging the gap between current feasibility and the state-of-the-art potential limited by resource constraints. The project effectively lays the groundwork for future advancements, highlighting areas where further investment could yield substantial improvements in AI-driven literary experiences.



SECTION 11: CONCLUSION

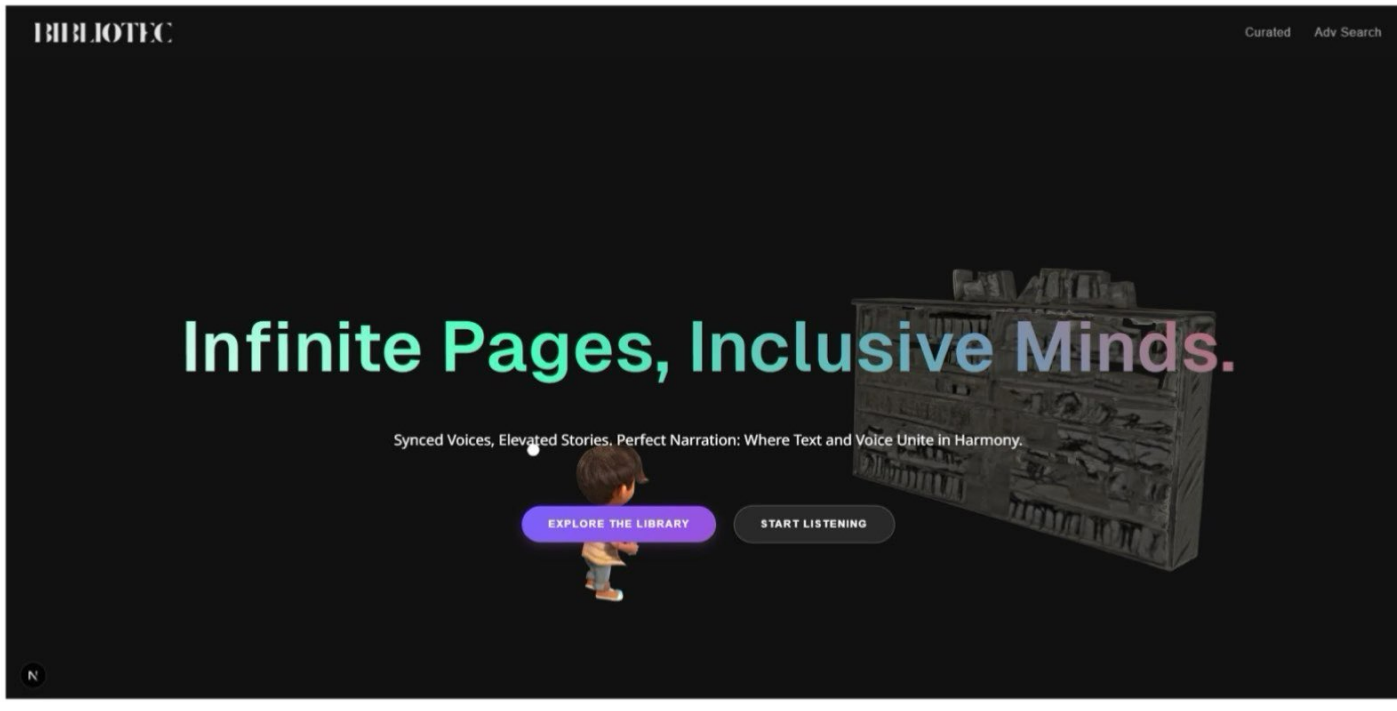
This project, initially conceived to address a personal desire for an enhanced literary engagement tool, successfully culminated in the development and deployment of the Bibliotec web platform (bibliotec.site). The core aim—to create a functional system synchronizing AI-generated audio with corresponding text passages—was achieved, delivering a multi-modal reading experience. Key technical achievements include the implementation of an automated content ingestion and processing pipeline, integration of a pre-trained TTS model (Kokoro TTS) modulated for basic expressiveness, and deployment via Azure cloud infrastructure, complete with browsing and search functionalities.

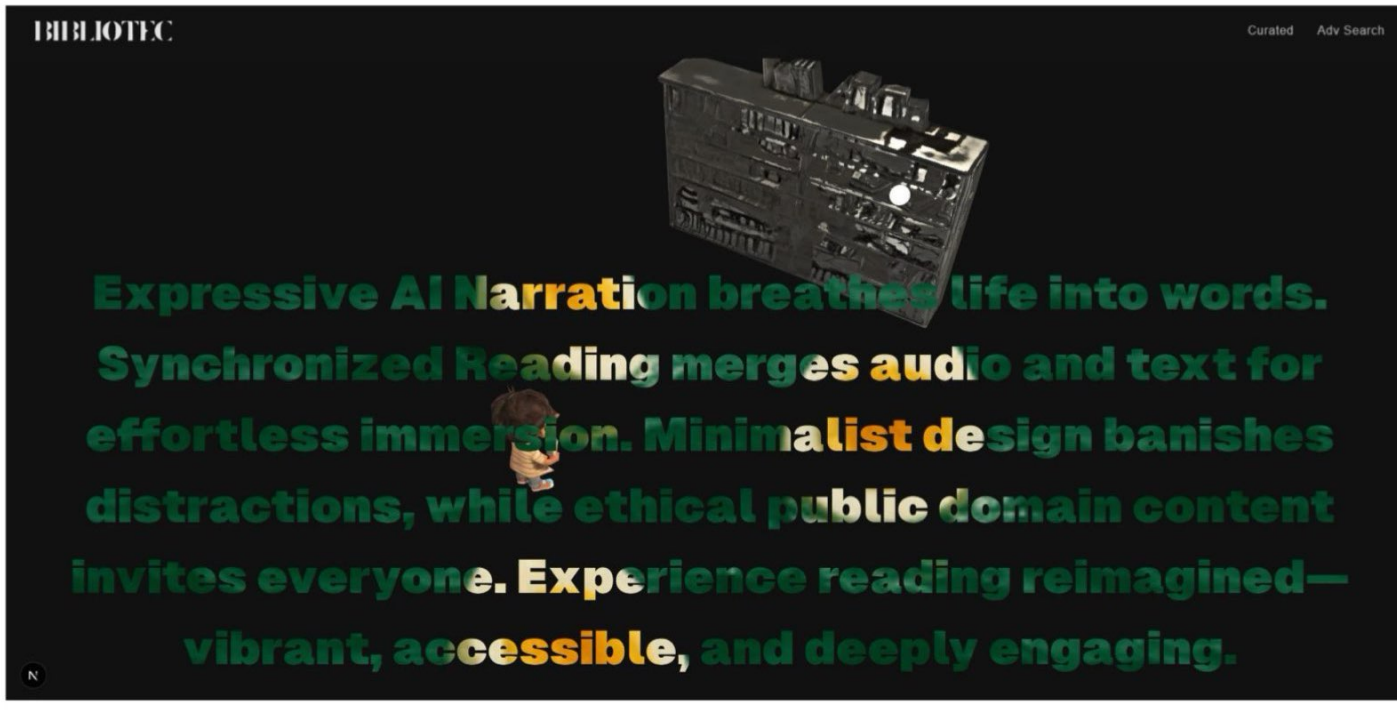
Crucially, user testing conducted with a diverse group of participants confirmed the platform's potential and broader appeal beyond the initial personal scope. Feedback was overwhelmingly positive, with users highlighting the intuitive design and the highly engaging nature of the synchronized audio-text feature, noting its effectiveness in maintaining focus and enhancing immersion. Participants expressed significant enjoyment using the platform and underscored its accessibility benefits, particularly for those favouring auditory learning or facing challenges with traditional reading. This positive reception validates the core concept and design choices.

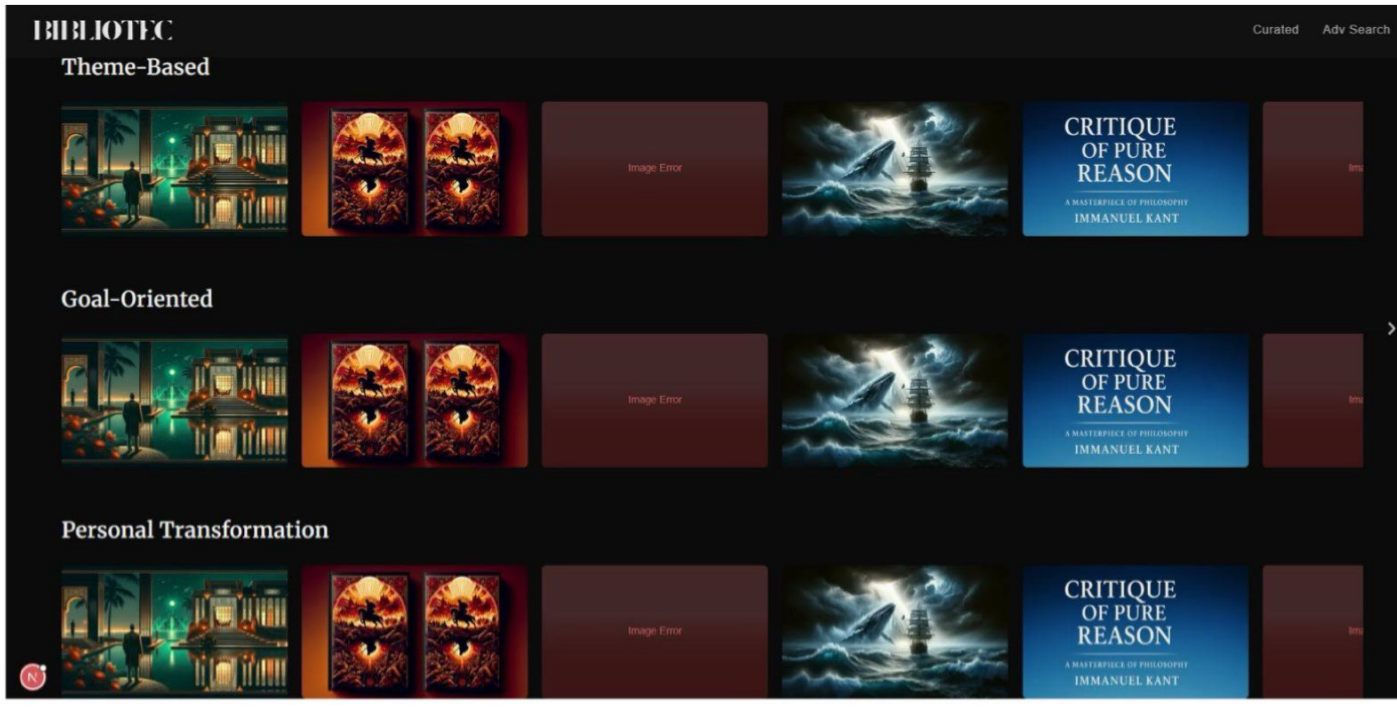
Following the platform's functional success and positive user feedback, it is important to contextualize the AI narration's expressiveness. The implemented approach, utilizing heuristic control on a pre-trained model, was a pragmatic necessity driven by practical constraints on computational resources and dataset availability. While this method does not achieve the full granularity of nuanced emotional delivery envisioned in the theoretically explored N-dimensional emotion model, it proved highly effective. As detailed earlier (Section 6.3), this resource-efficient strategy successfully delivered narration of very good quality and expressiveness, achieving significant processing speed (~10 hours of audio per compute hour) and meeting the core requirements for an engaging user experience.

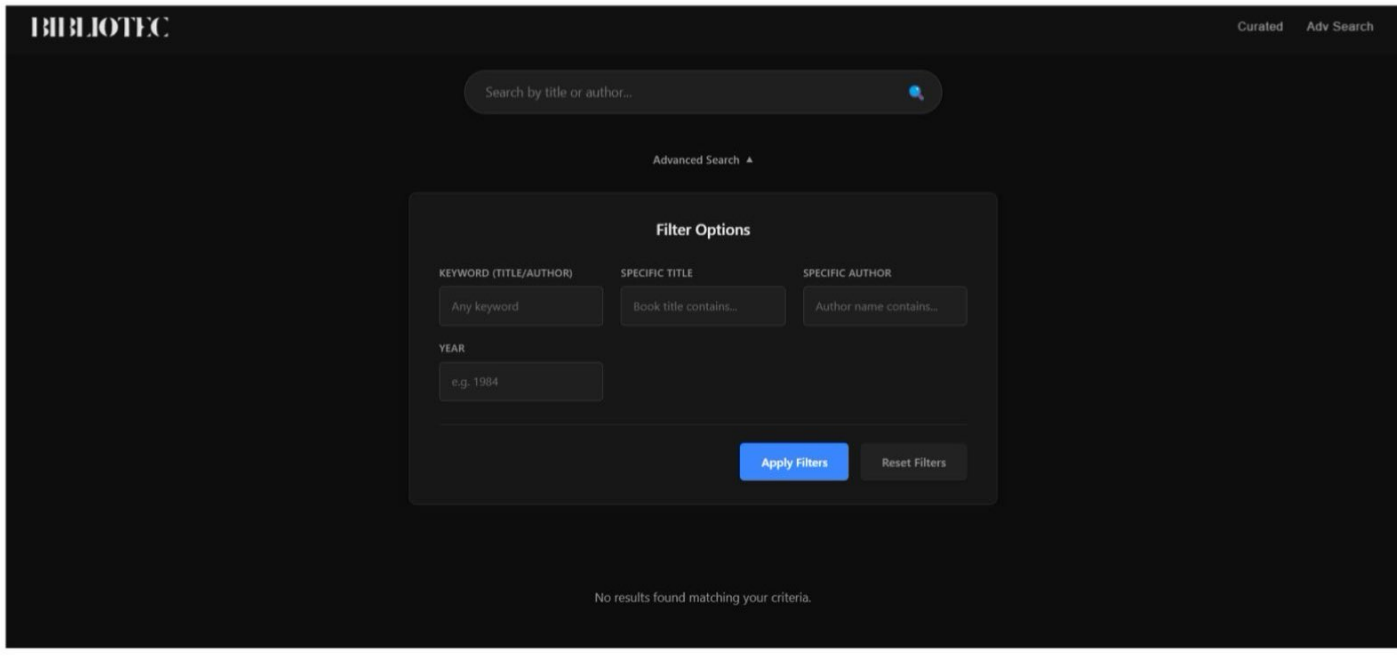
Nevertheless, the theoretical exploration into advanced TTS architectures and the N-dimensional emotion space provided valuable insights. It confirms the underlying principles for achieving superior results and demonstrates the significant potential for realizing a highly natural and emotionally resonant TTS system, should sufficiently time, data, and computational power become available in the future. This foundational research serves as a valuable roadmap for future development, complementing the successful and efficient practical implementation achieved within this project.

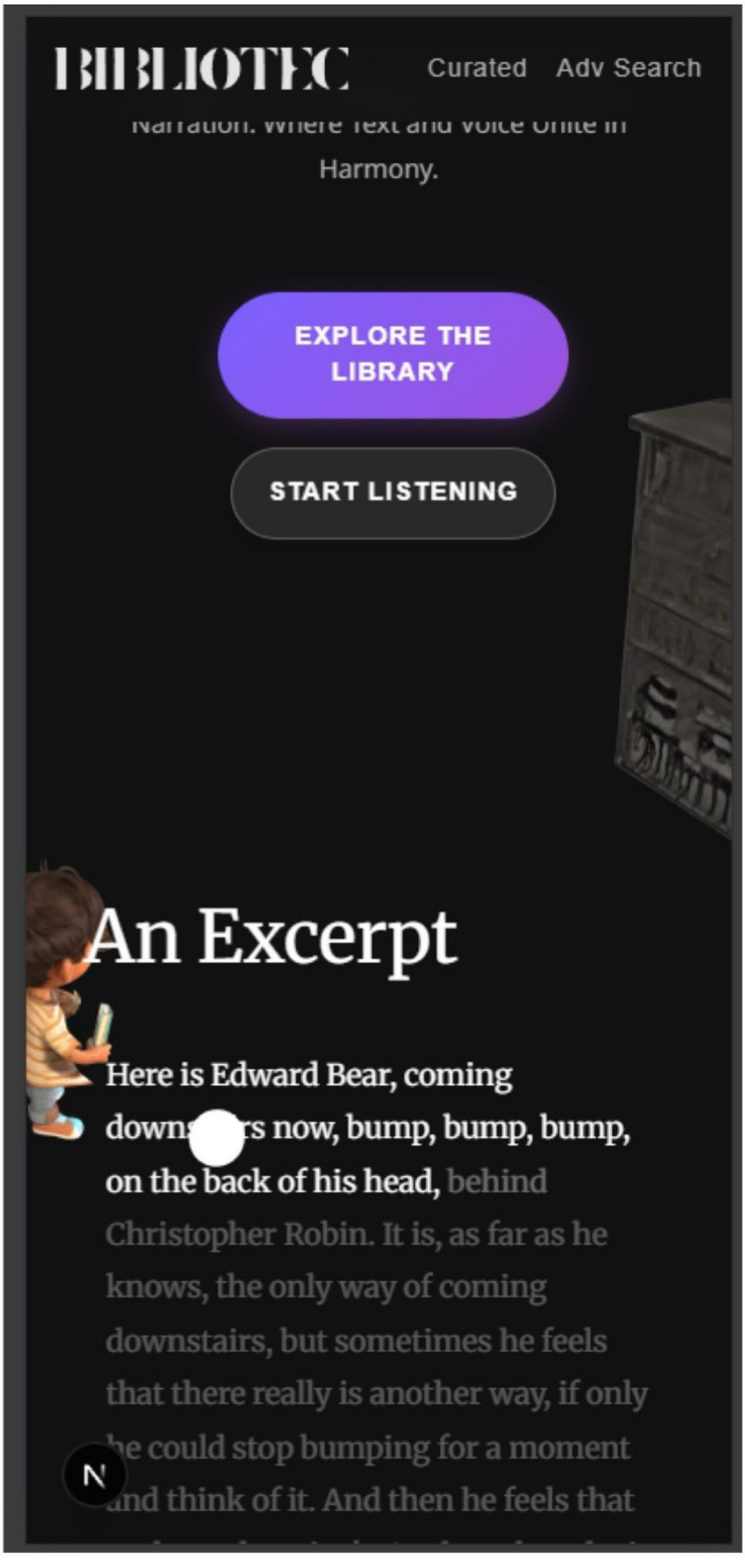
In conclusion, the Bibliotec project serves as a successful proof-of-concept. It not only met its primary objectives by delivering a functional and personally effective tool but also demonstrated significant potential for wider user engagement and accessibility, as confirmed by positive user testing. It showcases the practical application of TTS and web technologies while realistically acknowledging the gap between current feasibility and the state-of-the-art potential limited by resource constraints. The project effectively lays the groundwork for future advancements, highlighting areas where further investment could yield substantial improvements in AI-driven literary experiences.

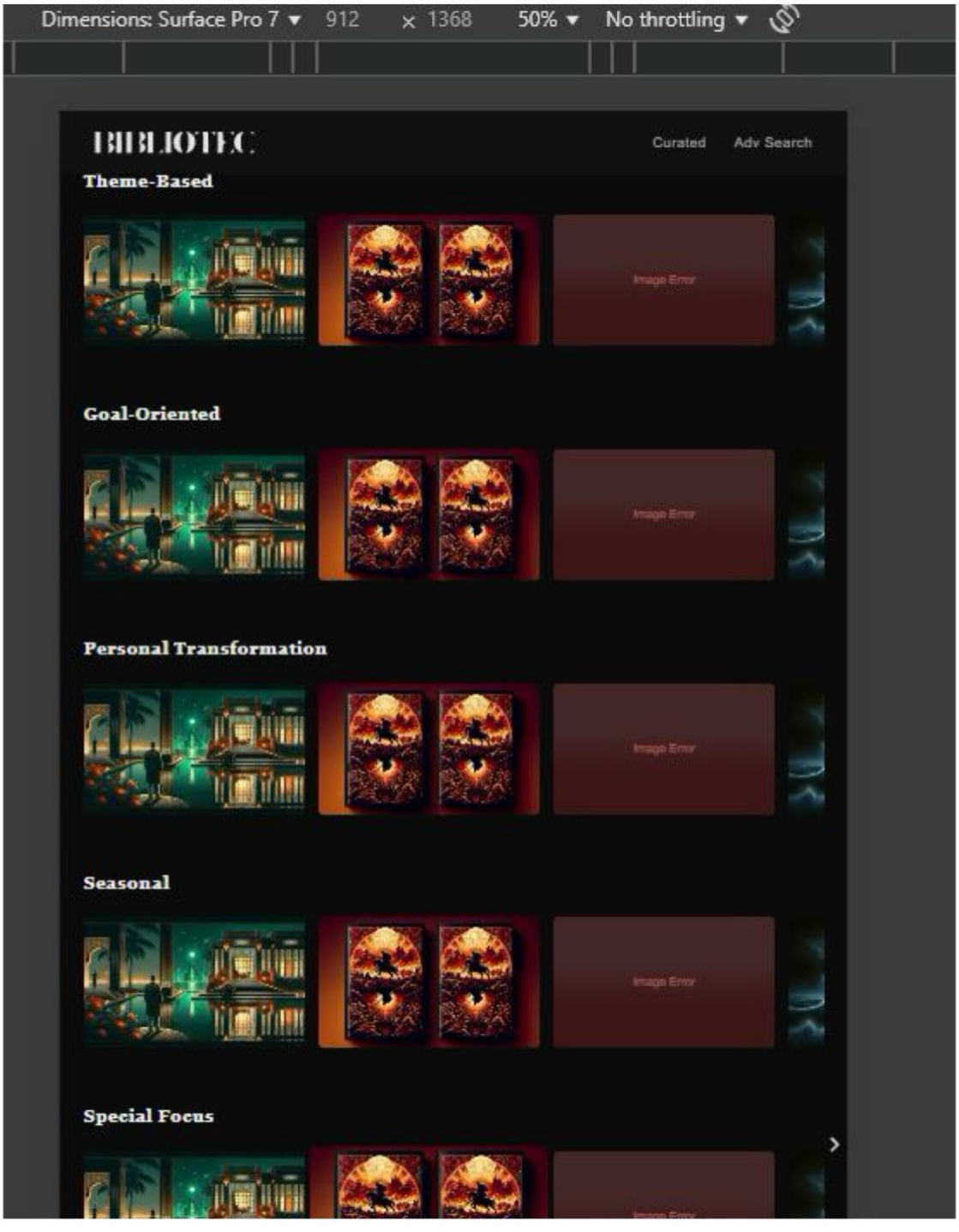

BIBLIOTEC
Curated
Adv Search
Infinite Pages, Inclusive Minds.
Synced Voices, Elevated Stories. Perfect Narration: Where Text and Voice Unite in Harmony.
EXPLORE THE LIBRARY
START LISTENING


BIBLIOTEC
Curated
Adv Search
Expressive AI Narration breathes life into words. Synchronized Reading merges audio and text for effortless immersion. Minimalist design banishes distractions, while ethical public domain content invites everyone. Experience reading reimagined—vibrant, accessible, and deeply engaging.


BIBLIOTEC
Curated
Adv Search
Theme-Based
Image Error
CRITIQUE OF PURE REASON
A MASTERPIECE OF PHILOSOPHY
IMMANUEL KANT
Goal-Oriented
Image Error
CRITIQUE OF PURE REASON
A MASTERPIECE OF PHILOSOPHY
IMMANUEL KANT
Personal Transformation
Image Error
CRITIQUE OF PURE REASON
A MASTERPIECE OF PHILOSOPHY
IMMANUEL KANT


BIBLIOTEC
Curated
Adv Search
Search by title or author...
Advanced Search
Filter Options
KEYWORD (TITLE/AUTHOR)
Any keyword
SPECIFIC TITLE
Book title contains...
SPECIFIC AUTHOR
Author name contains...
YEAR
e.g. 1984
Apply Filters
Reset Filters
No results found matching your criteria.


BIBLIOTEC
Curated
Adv Search
Harmony.
EXPLORE THE LIBRARY
START LISTENING
An Excerpt
Here is Edward Bear, coming
on the back of his head, behind
Christopher Robin. It is, as far as he
knows, the only way of coming
downstairs, but sometimes he feels
that there really is another way, if only


Dimensions: Surface Pro 7 912 x 1368 50% No throttling
BIBLIOTEC
Curated
Adv Search
Theme-Based
Image Error
Goal-Oriented
Image Error
Personal Transformation
Image Error
Seasonal
Image Error
Special Focus
Image Error

References

Baccianella, S., Esuli, A., & Sebastiani, F. (2010). SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining. In Proceedings of the Seventh Conference on International Language Resources and Evaluation (LREC'10). Retrieved from http://sentiwordnet.isti.cnr.it/

Batliner, A., Fischer, K., Huber, R., Spilker, J. and Nöth, E. (2006). 'Emotional speech processing'. In Acoustics, Speech, and Signal Processing, 2006. ICASSP 2006. IEEE International Conference on. IEEE, pp. 1013–1016.

Bird, S., Klein, E., & Loper, E. (2009). Natural Language Processing with Python. O'Reilly Media Inc.

Boersma, P., & Weenink, D. (2024). Praat: doing phonetics by computer (Version 6.3) [Computer program]. Retrieved from http://www.praat.org/

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). On the properties of neural machine translation: Encoder-decoder approaches. arXiv preprint arXiv:1409.1259.

Communeify. (2025, January 15). Kokoro TTS: Lightweight Open-Source Text-to-Speech Model - Complete Guide. Retrieved from https://www.communeify.com/en/blog/kokoro-tts-lightweight-open-source-text-to-speech-model-complete-guide

Eyben, F., Schuller, B., Batliner, A., Sirko, S., Eny, C., Zaika, M., Wollmer, M. and Reiter, F. (2016). 'Geneva minimalistic acoustic parameter set (GeMAPS) for voice research and technology'. IEEE Transactions on Affective Computing, 7(2), pp. 190–201.

Fontaine, J.R.J., Scherer, K.R., Roesch, E.B. and Ellsworth, P.C. (2007). The world of emotions is not two-dimensional. Psychological Science, 18(12), pp. 1050-1057. (Note: Original entry was for a book "The world of emotions", but the citation likely refers to dimensional models discussed in this paper or similar works by these authors. Adjusted based on common citation patterns.)

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.

Hexgrad. (n.d.). Kokoro-82M [Model card]. Hugging Face. Retrieved from https://huggingface.co/hexgrad/Kokoro-82M

Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint arXiv:1207.0580.

Jung, W., & Lee, J. (2023). E3-VITS: An End-to-End Emotional TTS System with a Language Model Conditioned by Both Reference Speech or Textual Description. arXiv preprint arXiv:2310.04037.

Juslin, P. N., & Laukka, P. (2003). Communication of emotions in vocal expression and music performance: Different channels, same code? Psychological Bulletin, 1 129(5), pp. 770–814. 2
kirj.ee / annales.umcs.lublin.pl

Lee, L., & Tashev, I. (2020). Improving Emotion Control in Neural Speech Synthesis Using Fine-Grained Conditioning. arXiv preprint arXiv:2006.05666.

Lee, S., Yildirim, S., Bulut, M., Kazemzadeh, A., Deng, Z., & Narayanan, S. (2004). An acoustic study of emotions expressed in speech. In ISCA archive (pp. 1041-1044).

Mauch, M., & Dixon, S. (2014). pYIN: A fundamental frequency estimator using probabilistic threshold distributions. In 2014 IEEE international conference on acoustics, speech and signal processing (ICASSP) (pp. 659-663). IEEE.

McAuliffe, M., Socolof, M., Mihuc, S., Wagner, M., & Sonderegger, M. (2017). Montreal Forced Aligner: Trainable Text-Speech Alignment Using Kaldi. In INTERSPEECH (pp. 498–502). DOI: 10.21437/Interspeech.2017-1386

McFee, B., Raffel, C., Liang, D., Ellis, D. P. W., McVicar, M., Battenberg, E., & Nieto, O. (2015). librosa: Audio and music signal analysis in python. In Proceedings of the 14th python in science conference (Vol. 8, pp. 18-25).

Mu, Z., Yang, X. and Dong, Y. (2021). 'Review of end-to-end speech synthesis technology based on deep learning'. arXiv preprint arXiv:2104.09995.

Pitrelli, J. F., Bakis, R., & Hirschman, L. (2006). Automatic detection of news story sentiment in spoken text. In Ninth international conference on spoken language processing (pp. 333-336).

Rabiee, H.R., Gotoh, Y., Suzuki, M. and Zhang, C. (2019). Adjusting Pleasure-Arousal-Dominance for Continuous Emotional Text-to-speech Synthesizer. In INTERSPEECH (pp. 3348-3352).

Robert, J., & Webber, J. (2011). Pydub (v0.24.1) [Python software]. Retrieved from http://pydub.com/

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. Nature, 323(6088), pp. 533-536.

Russell, J.A. (1980). A circumplex model of affect. Journal of personality and social psychology, 39(6), p.1161.

Scherer, K. R. (1986). Vocal affect expression: A review and a model for future research. Psychological Bulletin, 99(2), pp. 143–165.

Scherer, K.R., Johnstone, T. and Klasmeyer, G. (2003). Vocal expression of emotion. In Scherer, K. R., Schorr, A., & Johnstone, T. (Eds.), Handbook of affective sciences (pp. 433-456). Oxford University Press.

Shen, J., Pang, R., Weiss, R. J., Schuster, M., Jaitly, N., Yang, Z.,... & Wang, Y. (2018). Natural TTS synthesis by conditioning wavenet on mel spectrogram predictions. In ICASSP 2018 - 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 4779-4783). IEEE.

Taylor, P. (2009). Text-to-Speech Synthesis. Cambridge University Press.

Volkow, N. D., Wang, G.-J., Fowler, J. S., Tomasi, D. and Telang, F. (2011). 'Addiction: beyond dopamine reward circuitry'. Proceedings of the National Academy of Sciences, 108(37), pp. 15037–15042. doi: 10.1073/pnas.1010654108.

Wang, Y., Skerry-Ryan, R. J., Stanton, D., Yin, Y., Jaitly, N., Weiss, R. J.,... & Xiao, Z. (2017). Tacotron: Towards end-to-end speech synthesis. arXiv preprint arXiv:1703.10135.

Wilmer, H. H., Sherman, L. E. and Chein, J. M. (2017). 'Smartphones and Cognition: A Review of Research Exploring the Links between Mobile Technology Habits and Cognitive Functioning'. Frontiers in Psychology, 8, p. 605. doi: 10.3389/fpsyg.2017.00605.

Yildirim, S., Bulut, M., Lee, C.M., Kazemzadeh, A., Deng, Z., Lee, S., Narayanan, S. and Busso, C. (2004). An acoustic study of emotions expressed in speech. In ISCA Archive.

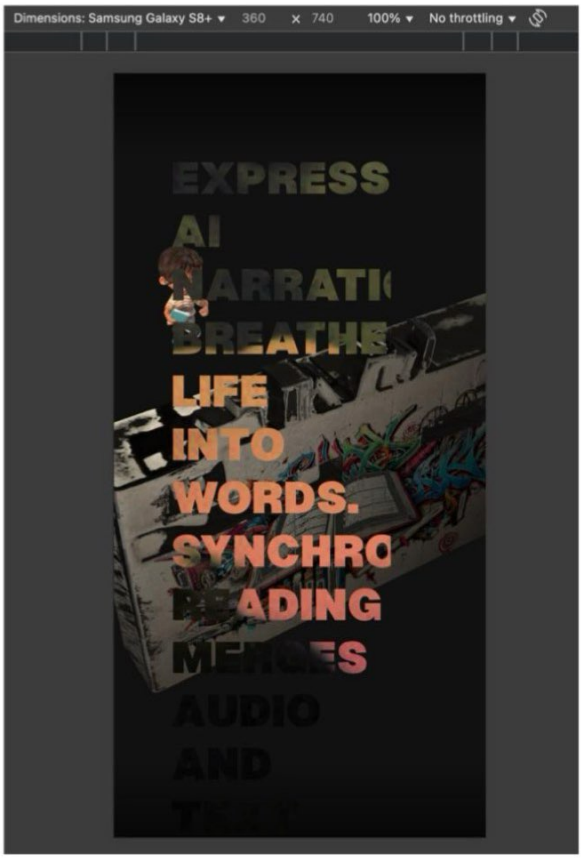
APPENDIX

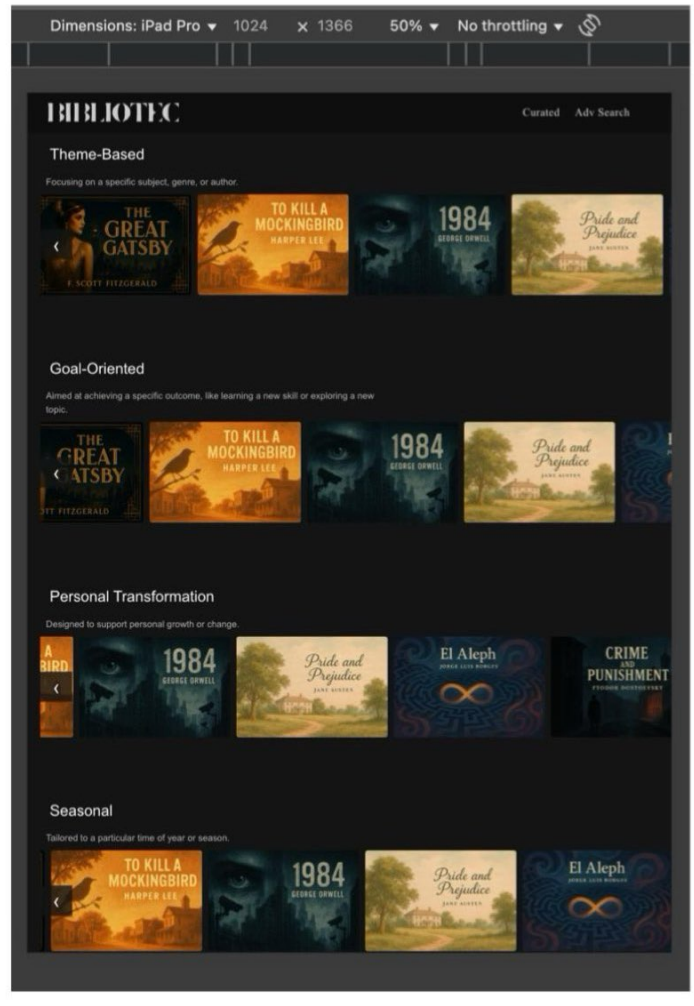
Meetings

Group meeting: approved for project
Supervisor meeting: discussed project ideas, if the philosophy if it suits my-self needs, user testing is not needed.
28/10/24 supervisor meeting : supervisor meeting discussing ways to research, write. Asked regarding ethical issues, the solution lies in what the public thinks.

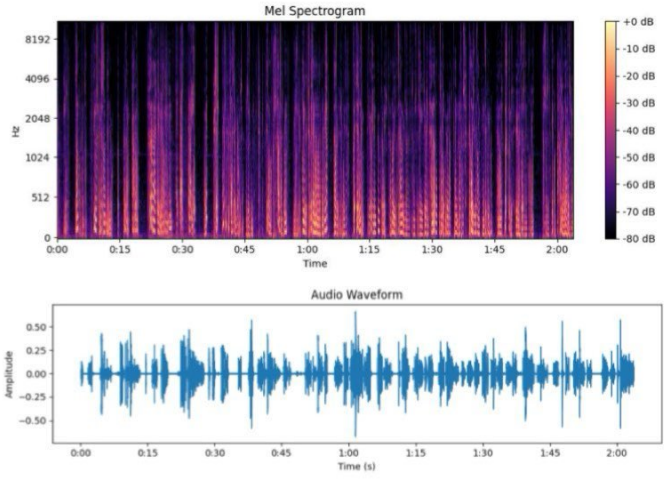
9/4/25: Supervisor meeting discussed little

Website screenshots showcasing accessibility for various devices




Dimensions: iPad Pro 1024 × 1366 50% No throttling
BIBLIOTEC
Curated
Adv Search
Theme-Based
Focusing on a specific subject, genre, or author.
THE GREAT GATSBY
TO KILL A MOCKINGBIRD
1984
Pride and Prejudice
Goal-Oriented
Aimed at achieving a specific outcome, like learning a new skill or exploring a new topic.
Personal Transformation
Designed to support personal growth or change.
El Aleph
CRIME AND PUNISHMENT
Seasonal
Tailored to a particular time of year or season.

TTS demonstration – final implementation of Bibliotec's TTS



TTS of training stages